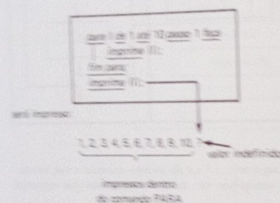
Observe que:

1) Assim como no comando enquanto, se o valor inicial (i) já for superior ao limite (l), os comandos $C_1, C_2, \ldots, C_n$ não serão executados.

2) Toda vez que o fim para é encontrado, a variável de controle (i) é incrementada pelo passo (p) e o teste (v < l) é feito.

3) O valor da variável de controle torna-se indefinido assim que é executado o comando para. Neste aspecto o comando para é diferente do comando enquanto. Por exemplo, com o trecho de algoritmo abaixo:



Quando o passo (p) for igual a 1, não será necessário escrever esta especificação no comando. No exemplo acima teríamos:

```
para I de 1 até 10 faça
    imprima (I);
fim para;
```

Exemplos:

O que será impresso no algoritmo abaixo?

```
a) início
     inteiro: I;
     para I de 1 até 10 passo 2 faça
         imprima (I);
     fim para;
   fim.
```



```
b) início
     inteiro: I, J;
     para I de 1 até 3 passo 1 faça
         para J de 1 até 5 passo 1 faça
             imprima (I, J);
         fim para;
     fim para;
   fim.
```

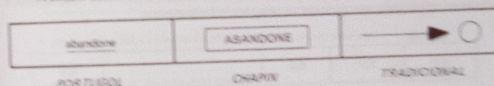
Soluções:

a) 1, 3, 5, 7, 9

b) 1 1, 1 2, 1 3, 1 4, 1 5
   2 1, 2 2, 2 3, 2 4, 2 5
   3 1, 3 2, 3 3, 3 4, 3 5

4.3 O COMANDO ABANDONE

A sintaxe deste comando é:



O comando abandone só tem sentido dentro de um comando de repetição (enquanto, repita, para). Além disso, estará sempre associado ao teste de uma condição com comando se.

A semântica do comando é a seguinte: quando o abandone é encontrado, o próximo comando a ser executado é o primeiro comando logo após o fim do comando de repetição mais interno onde aparece o abandone.

Exemplo:

